AMENDMENTS TO THE CLAIMS

1. (currently amended) A method of performing memory mapped input output operations to an alternate address space comprising:

establishing a first instruction directed to a first memory mapped input output alternate address space associated with an adapter to store data in accordance with ~~a definition of a z/Architecture~~ resource address designation, said resource address designation configured for decomposition thereof such that said first memory mapped input output alternate address space associated with said adapter is accessible;

establishing a second instruction directed to said first memory mapped input output alternate address space associated with an adapter to load data in accordance with ~~a definition of a z/Architecture~~ said resource address designation;

allocating, through a host program, at least one of a real resource and a virtual resource associated with said first memory mapped input output alternate address space to a ~~process~~ guest program started by the host program;

ensuring that ~~said selected~~ a process executed by the guest program corresponds ~~with said process~~ to ~~which~~ said resource ~~is~~ allocated to the guest program, in a manner that is not visible to the guest program; and

wherein said process issues at least one of said first instruction and said second instruction and thereby causes execution of at least one of said store and load with said first memory mapped input output alternate address space.

2. (currently amended) The method of Claim 1 further ~~including~~ comprising allocating, through the host program on behalf of the guest program, an ~~error storage area associated with an alternate address space that is further associated with an~~ error storage area~~, storage area type, fetched by execution of said second instruction~~ associated with said resource.

3. (currently amended) The method of Claim 1 further including virtualization of

a resource of one of said adapter to store data and said adapter to load data to a second-level guest process.

4. (currently amended) The method of Claim 3 wherein said virtualization of a resource is accomplished and distinguished from a real resource by partitioning a range of resource identifiers into a plurality of portions;

wherein at least one portion corresponds to a virtual resource; and

wherein when at least one of said first instruction and said second instruction specifies a resource identifier corresponding to said at least one portion, the guest program issuing ~~said~~ instruction exits, and an ~~underlying~~ the host program resumes execution in order to emulate said at least one of said first instruction and said second instruction originally issued by the guest program.

5. (currently amended) The method of claim 3 wherein said virtualization provides direct access to at least one of a real resource and a virtual resource of an adapter by a problem-state second-level guest process.

6. (currently amended) The method of claim 5 wherein said access is accomplished without involvement from a kernel of ~~said~~ a guest operating system; and permits said process operating in a problem-state maximum efficiency in performing the primary input output capabilities provided by said adapter and the associated resources allocated to said process.

7. (currently amended) The method of Claim 3 further including separating another process operating under ~~said~~ an operating system; wherein said separating is established on a per-resource basis during said allocating and is enforced during execution of at least one of said first and said second instructions.

8. (original) The method of Claim 1 wherein said first alternate address space is not a portion of the main address space from which said process is executing.

9. (currently amended) The method of Claim 1 wherein said process issuing said at least one of said first instruction and said second instruction and thereby causes execution of at least one of said store and load with said first alternate address space operates in a problem state of a machine.

10. (cancelled)

11. (cancelled)

12. (original) The method of Claim 1 wherein at least one of said first instruction and said second instruction is executed without supervisory state intervention.

13. (currently amended) The method of Claim 1 wherein said first instruction and said second instruction are semiprivileged instructions that may be executed in a problem state, wherein ownership of a specified resource of a specified adapter determines a privilege required for execution of said semiprivileged instructions.

14. (currently amended) The method of Claim 1 further including a second memory mapped input output alternate address space associated with a second adapter.

15. (currently amended) The method of Claim 14 wherein a storage location in said first memory mapped input output alternate address space maps to a different address than the same location in said second memory mapped input output alternate address space.

16. (currently amended) The method of Claim 1 wherein said adapter includes address spaces as partitions of said alternate address space.

17. (currently amended) The method of Claim 1 wherein an address space is governed by at least one of a resource type and storage area types associated with ~~an~~ said adapter.

18. (currently amended) A storage medium encoded with a ~~machine~~computer-readable computer program code, said code including instructions ~~for causing~~that, when executed, cause a computer to implement a method of performing memory mapped input output operations to an alternate address space, the method comprising:

establishing a first instruction directed to a first memory mapped input output alternate address space associated with an adapter to store data in accordance with a definition of a ~~z/Architecture~~resource address designation, said resource address designation configured for decomposition thereof such that said first memory mapped input output alternate address space associated with said adapter is accessible;

establishing a second instruction directed to said first memory mapped input output alternate address space associated with an adapter to load data in accordance with ~~a definition of a z/Architecture~~ said resource address designation;

allocating, through a host program, at least one of a real resource and a virtual resource associated with said first memory mapped input output alternate address space to a ~~process~~ guest program started by the host program;

ensuring that ~~said selected a~~ process executed by the guest program corresponds ~~with said process~~ to ~~which~~ said resource ~~is~~ allocated to the guest program, in a manner that is not visible to the guest program; and

wherein said process issues at least one of said first instruction and said second instruction and thereby causes execution of at least one of said store and load with said first memory mapped input output alternate address space.

19. (currently amended) A system for performing memory mapped input output operations to an alternate address space comprising:

a means for establishing a first instruction directed to a first memory mapped input output alternate address space associated with an adapter to store data in accordance with a ~~definition of a z/Architecture~~resource address designation, said resource address designation configured for decomposition thereof such that said first memory mapped input output alternate address space associated with said adapter is accessible;

a means for establishing a second instruction directed to said first memory mapped input output alternate address space associated with an adapter to load data in accordance with ~~a definition of a z/Architecture~~ said resource address designation;

a means for allocating, through a host program, at least one of a real resource and a virtual resource associated with said first memory mapped input output alternate address space to a ~~process~~ guest program started by the host program;

a means for ensuring that ~~said selected~~ a process executed by the guest program corresponds ~~with said process~~ to ~~which~~ said resource ~~is~~ allocated to the guest program, in a manner that is not visible to the guest program; and

wherein said process issues at least one of said first instruction and said second instruction and thereby causes execution of at least one of said store and load with said first memory mapped input output alternate address space.